NASA-TM-84633 19830012319

NASA Technical Memorandum 84633

TEKLIB GRAPHICS LIBRARY

Susan W. Bostic

MARCH 1983

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23665

TABLE OF CONTENTS

SUMMARY

The Finite Element Machine is a parallel processor designed to solve structural analysis problems. It consists of a TI 990, which acts as a controller, and an array of 36 microcomputers connected by specially designed hardware. TEKLIB is a library of procedures designed to provide an interface between the TI 990 and an interactive terminal, to enable the user to input data graphically and to display output from the FEM.

INTRODUCTION

TEKLIB is a library of procedures written in TI PASCAL to perform basic graphic tasks. The library was written specifically to provide an interface between a Tektronix 4014 or equivalent and the TI 990. The TI 990 is used as a controller for the Finite Element Machine which is an array of 36 TI 9900 microprocessors designed to solve problems by finite element methods in parallel. The use of the TEKLIB procedures provides a means of inputting data graphically and displaying output.

Because the library was written as a special purpose tool, some procedures are computer-dependent and some use features exclusive to TI PASCAL. However, the library could be modified for future additions and modifications to the system and it is anticipated that more procedures will be added as the need indicates.

TEKLIB was modelled after TKLIB, a Fortran graphics library package available at Langley under the NOS and PRIME 300 systems.

OVERVIEW

TEKLIB is a procedure callable library of two-dimensional graphic routines stored on the FEM disk on file FEM.SWP.TEK.LIB. Before using these procedures, the user should be familiar with certain features described below. These include: common block description, buffer and file management, logical and screen coordinate relationships, and display options.

A brief description of each procedure will be found followed by a simple short example problem.

1. Common Block Description

Common and access declarations are a special feature of TI PASCAL. They are used to declare variables which may be shared with other routines falling within the scope of the common declaration or with externally compiled routines.

Every program which uses TEKLIB must include the following common declaration. Every routine which accesses any of the common variables must have an access declaration.

The following common declaration is stored in FEM.SWP.TEK.MANUAL.COMMON:

```
COMMON G:  RECORD
    ENTRY : BOOLEAN;                          (*   SET TO TRUE INITIALLY   *)
    XMINL,XMAXL,YMINL,YMAXL: REAL             (*   LOGICAL COORDINATES     *)
    XMIN,XMAX,YMIN,YMAX : REAL;               (*   PHYSICAL COORDINAGES    *)
    XORG,YORG,XSCALE,YSCALE: REAL;            (*    ORIGINS AND SCALES     *)
    ICX,ICY : INTEGER;                        (* CURRENT RASTER POSITION   *)
    PTR : INTEGER;                            (* CURRENT BUFFER POSITION   *)
    TRAN : (SAVEON,SENDON,BOTH);              (*  SAVE, SEND, BOTH         *)
    MODE : (GRAPHIC,ALPHA);                   (* 29 GRAPHIC, 31 ALPHA      *)
    BEAM : (ON,OFF);                          (* ON FOR DRAW, OFF FOR MOVE *)
    INTENSITY: INTEGER;                       (*    BEAM INTENSITY         *)
    DIRECTION : (UP,DOWN,LEFT,RIGHT);         (* CURSOR DIRECTION          *)
    SIZE : (LARGE,MEDIUM,SMALL,XSMALL);       (* SYMBOL SIZE               *)
    LINETYPE : (SOLID,DOTTED,DOTDASH,SHORTDASH,LONGDASH); (*TYPE           *)
    ERROR : INTEGER;                          (*    ERROR CODE             *)
    LBUFF: INTEGER;                           (* LENGTH BUFFER DIMENSIONED *)
    BUFF : ARRAY[1..80] OF INTEGER;           (*    USER BUFFER            *)
    TBUFF: ARRAY[1..40] OF INTEGER;           (*   TRANSMISSION BUFFER     *)
    END;
```

The above common variables are described in the following sections.

2. Buffer and File Management

The common variable G.BUFF defines an area that is set aside as a buffer in which to store the graphics data. The length of the buffer is limited to 80 characters as only 80 characters of data can be sent at a time. The use of this buffer, instead of transmitting on a vector to vector basis, increases the speed and efficiency of drawing images.

The value of the common variable G.TRAN determines when to store data in the user's buffer and also when to save the data on a save file. The value of G.TRAN is set within the graphics procedures. When these default values are used, the use of the user's buffer is transparent and the graphics data is automatically written to a graphics output file labelled "GROUT". At the end of a session this file should be saved under the user's directory. If the default conditions do not suit the user, further information on the effects of changing the value of G.TRAN can be found in the Appendix.

If the user wishes to rewrite the save file the value of the common variable G.ENTRY should be set to TRUE. This will write over what has already been written.

The common variable G.PTR is the current index of the user's buffer. G.PTR is always pointing to the next available space in the buffer, so that at any point in time, there are (G.PTR-1) words in the buffer.

G.ENTRY	Initializes the SAVE file. Set to TRUE initially by TKSTUP.
G.BUFF	An array which will hold up to 80 characters to be transmitted.
G.PTR	The current index of the user's buffer array.
G.TBUFF	An array of 40 words reconstructed for transmittal to TTY.
G.LBUFF	Set to 80 in TKSTUP. Length of user's buffer.
G.TRAN	The variable which controls whether to save the command, send the command directly or to save and send.

3. Coordinate Definition

This library of procedures refers to two types of coordinate systems. The 'PHYSICAL' coordinates define the addressable points on the screen. The 'LOGICAL' coordinates are in the user's units, such as inches, meters, miles, etc.

The following common variables are a function of the physical screen size, in this case 1024X by 780Y. Space is set aside outside the graphics area for queries and annotation. The actual physical size of the graphic area is 650 by 650.

G.XMIN	Set to 300 in TKSTUP. Minimum physical coordinate of X
G.XMAX	Set to 950 in TKSTUP. Maximum physical coordinate of X
G.YMIN	Set to 100 in TKSTUP. Minimum physical coordinate of Y
G.YMAX	Set to 750 in TKSTUP. Maximum physical coordinate of Y
G.ICX	Current raster position in the X direction
G.ICY	Current raster position in the Y direction

The basic procedures use physical coordinates (i.e., screen coordinates within range of 0X to 1024X and 0Y to 780Y) as opposed to the logical values of the user. The procedures TKLTP and TKPTL convert these values to logical or physical coordinates, respectively.

The user must input the following values:

G.XMINL	The logical minimum value of X
G.XMINL	The logical minimum value of Y
G.XMAXL	The logical maximum value of X
G.YMAXL	The logical maximum value of Y

Given these values the procedure TKSCA will compute G.XORG and G.YORG, the physical coordinates of the 'LOGICAL' origin, and G.XSCALE and G.YSCALE, the raster units per user's unit.

Example

Given:

G.XMINL = 0.	G.YMINL = 100.
G.XMAXL = 50.	G.YMAXL = 400.

TKSCA returns values of:

G.XORG = 300.0	G.YORG = -116.667
G.XSCALE= 13.0	G.YSCALE= 2.16667

When using TKGRID or TKGRD (the grid generation routines) the following values must be input:

NDIVX	The number of divisions in X direction
NDIVY	The number of divisions in Y direction

4. Display options

The following common variables are used to control the graphic display:

G.DIRECTION	Specifies direction to move cursor
G.SIZE	Specifies size of characters to be displayed
G.LINETYPE	Specifies type of lines to be displayed
G.BEAM	Specifies whether beam is on or off
G.INTENSITY	Specifies intensity of beam
G.MODE	Specifies graphic or alpha mode

The use of each of the above variables is described in the writeups of the procedures in which they are used.

TKSTUP

Purpose: To initialize common block variables

Declaration: Procedure TKSTUP;

Parameters: None

Procedures Called: None

Description: Physical size of display area is defined. A display screen of 1024X by 780Y, like the Tektronix 4014 is assumed. The graphic display area is set up as 300 to 950 in X direction, and 100 by 750 in Y direction. This leaves room for annotation and queries.

The following variables are initialized:

```
G.ENTRY:= TRUE;
G.SIZE  := MEDIUM;
G.PTR   := 1;
G.TRAN  := BOTH;
G.LBUFF := 80;
```

G.INTENSITY := 50;
G.BUFF and G.TBUFF arrays are set to zero.

Note: This procedure must be called before any others.

TKSCA

Purpose: Given XMINL,XMAXL,YMINL,YMAXL, compute values of X and Y at origin and X and Y scales and store in common block.

Declaration: Procedure TKSCA;

Parameters: None

Procedures Called: None

Description:

```
G.XSCALE := (G.XMAX-G.XMIN) / (G.XMAXL-G.XMINL);
G.YSCALE := (G.YMAX-G.YMIN) / (G.YMAXL-G.YMINL);
G.XORG := G.XMIN-G.XMINL*G.XSCALE;
G.YORG := G.YMIN-G.YMINL*G.YSCALE;
```

TKLTP

Purpose: To convert logical coordinates to physical coordinates.

Declaration: Procedure TKLTP(VAR XP,YP: INTEGER; XL,YL: REAL);

Parameters:

Input

XL,YL	Logical coordinates of point

Output

XP,YP	Physical coordinates of given point

Procedures Called: None

Description: Common variables XORG,YORG,XSCALE,YSCALE are obtained from a previous call to TKSCA. Physical values are computed as follows:

```
XP = G.XORG + G.XSCALE * XL
YP = G.YORG + G.YSCALE * YL
```

TKPTL

Purpose: Given XP,YP (physical coordinages)
Compute XL,YL (logical coordinates)

Declaration: Procedure TKPTL(VAR XL,YL: REAL; XP,YP: INTEGER);

Parameters:
  Input
    XP,YP Physical coordinates

  Output
    XL,YL Logical coordinates of given point

Procedures called: None

Description: Origins and scales are obtained from a previous call to TKSCA. Logical values are computed as follows:

```
XL = (XP-G.XORG)/G.XSCALE
YL = (YP-G.YORG)/G.YSCALE
```

TKCOORDS

Purpose: Given HIX,LOX,HIY,LOY
Compute the coordinates X and Y

Declaration: Procedure TKCOORDS(HIX,LOX,HIY,LOY,X,Y: INTEGER);

Procedures Called: None

Parameters:
  Input
    HIX,LOX,HIY,LOY The four data byte address

  Output
    X,Y The physical coordinates

Description: The coordinates are computed using following equations:

```
X := (HIX-32)*32 + (LOX-32);
Y := (HIY-32)*32 + (LOY-32);
```

TKHILO

Purpose: To convert a given X,Y to HIY,LOY,HIX,LOX data bytes of the graphic address.

Declaration: Procedure TKHILO(VAR X,Y,HIX,LOX,HIY,LOY:INTEGER);

Parameters:
  Input
    X,Y Physical coordinates of address

  Output
    LOY,HIY,LOX,HIX Data bytes of graphic address

Procedures Called: None

Description: A complete graphic address consists of four data bytes- HI Y, LO Y, HI X and LO X, received in that order. This procedure converts the X,Y physical address to the four byte graphic address.

TKSEND

Purpose: To send commands to Tektronix

Declaration: Procedure TKSEND(NWDS:INTEGER);

Parameters:
Input
NWDS Number of words to be sent

Procedures Called: SVC$

Description: This procedure is the one that actually transmits the data to the Tektronix. A supervisor call block is set up which includes the write ASCII command to LUNO #AD. The user's buffer is packed into the transmission buffer, the entire buffer is sent and the pointer is reinitialized. If there is a chance of more than 80 words in buffer, TKCHEK should be called first.

Maximum Buffer Words Sent: G.PTR-1

Note: This procedure is dependent on the DX10 operating system on the TI 990 controller.

TKERASE

Purpose: Erases screen
Clears buffer

Declaration: Procedure TKERASE;

Parameters: None

Procedures Called: TKSEND, DELAY

Description: The erase command is sent directly to terminal. The buffer pointer is reset to one. A delay of one second is executed.

Note: Mode is automatically set to alpha.

TKCURSOR

Purpose: To read the terminal's graphic cursor and the character used to signal the input

Declaration: Procedure TKCURSOR(VAR X,Y,CHARAC:INTEGER);

Parameters:
  Input None

  Output
    X,Y The physical coordinates of the cursor position
    CHARAC The numeric equivalent of the ASCII code of the character entered at the terminal

Procedures Called: SVC$

Description: When called, the procedure will enter a read phase and display the crosshair cursor on the screen. A character entered while cursor is displayed causes transmission of that character, followed by the four-byte address of the crosshair cursor.

Note: Procedure reads NWDS+1 to avoid leaving a CR in buffer

TKMOVE

Purpose: To move cursor to X,Y

Declaration: Procedure TKMOVE (X,Y:INTEGER);

Parameters:
  Input
    X,Y Physical coordinates of screen

Description: Cursor is moved invisibly from current position to the desired position. The current raster position is reset and the mode is set to graphic.

Maximum buffer words: 5

TKMDIST

Purpose: To move cursor X,Y distance from current position

Declaration: Procedure TKMDIST (X,Y:INTEGER);

Parameters:
  Input
    X Distance to move in X direction in physical dimensions
    Y Distance to move in Y direction in physical dimensions

Procedures Called: TKMOVE

Description: The cursor is moved invisibly the given distance in physical dimensions

Maximum Buffer Words: 5

TKMLDIS

Purpose: To move cursor X,Y logical distance from current position

Declaration: Procedure TKMLDIS (X,Y:INTEGER);

Parameters:
  Input
    X Distance to move in X direction in logical dimensions
    Y Distance to move in Y direction in logical dimensions

Procedures Called: TKMOVE, TKLTP

Description: The cursor is moved invisibly the given distance in logical dimensions

Maximum Buffer Words: 5

TKVECTOR

Purpose: To draw a vector from X1,Y1 to X2,Y2

Declaration: Procedure TKVECTOR (X1,Y1,X2,Y2:INTEGER);

Parameters:
  Input
    X1,Y1 Physical coordinates from which to draw
    X2,Y2 Physical coordinates to which to draw

Procedures Called: TKHILO, TKSEND

Description: The given coordinates are converted to their four-byte addresses, placed in buffer and sent according to value of G.TRAN. The current raster position is reset and the mode is set to graphic.

Maximum Buffer Words: 9

TKDM

Purpose: Move or draw to X,Y
Mode is set to graphic

Declaration: Procedure TKDM (X,Y:INTEGER);

Parameters:
  Input
    X,Y Physical coordinates of screen

Procedures Called: TKHILO, TKSEND

Description: Draws or moves from current position to desired position according to value of G.BEAM. If G.BEAM is equal to ON the cursor draws if equal to OFF the cursor is moved only.

Maximum Buffer Words: 5

TKDRAW

Purpose: Draw from current position to X,Y
Assumptions: Mode is already set to graphic and starting point is already in buffer

Declaration: Procedure TKDRAW(X,Y:INTEGER);

Parameters:
Input
X,Y Physical coordinates

Procedures Called: TKHILO, TKSEND

Description: TKDRAW loads new address into buffer. The assumptions are made that the starting point is already in buffer and that mode is set to graphic. The current raster position is updated.

Maximum Buffer Words: 4

TKDDIST

Purpose: To draw cursor X,Y distance from current position

Declaration: Procedure TKDDIST (X,Y:INTEGER);

Parameters:
Input
X Distance to draw in X direction in physical dimensions
Y Distance to draw in Y direction in physical dimensions

Procedures Called: TKDRAW

Description: A line is drawn the given distance in physical dimensions

Maximum Buffer Words: 5

TKDLDIS

Purpose: To draw cursor X,Y logical distance from current position.

Declaration: Procedure TKDLDIS (X,Y:INTEGER);

Parameters:
  Input
    X Distance to draw in X direction in logical dimensions
    Y Distance to draw in Y direction in logical dimensions

Procedures Called: TKDRAW

Description: A line is drawn the given distance in logical dimensions

Maximum Buffer Words: 5

TKREDRAW

Purpose: To redraw a given figure defined by nodes

Declaration: Procedure TKREDRAW(NNODES:INTEGER; NODE:ARRAY[1..3,1..?] of INTEGER);

Parameters:
  Input
    NNODES Number of values in node array
    NODE An array which contains node description
      where NODE[1,I] contains node number
            NODE[2,I] contains X coordinate
            NODE[3,I] contains Y coordinate

Procedures Called: TKVECTOR

Description: Vectors are drawn between given nodes. G.TRAN is set to both.

Maximum Buffer Words: NNODES * 8 +1

TKPOINT

Purpose: To draw a point

Declaration: TKPOINT(X,Y:INTEGER);

Parameters:
  Input
    X,Y Physical coordinates of point

Procedures Called: TKHILO,TKSEND

Description: Mode is set to graphic and special point plot option is entered into buffer, followed by the intensity character, the four data bytes of the address and a 29 to reset to graphic mode

Maximum Buffer Words: 7

Note: This routine assumes terminal has an enhanced graphic package which recognizes special point plot mode and intensity character.

TKAXES

Purpose: To draw a set of axes for display area

Declaration: Procedure TKAXES;

Procedures Called: TKVECTOR

Parameters: None

Description: X and Y axes are drawn for display area setup by TKSTUP

Maximum Buffer Words: 18

TKCHEK

Purpose: To check buffer overflow

Declaration: Procedure TKCHEK(NWDS:INTEGER);

Parameters:
  Input
    NWDS Number of words to be sent by next command

Procedures Called: TKSEND

Description: If the addition of NWDS to buffer would cause overflow, then (G.PTR-1) words already stored in buffer are sent and G.ERROR is set to 1.

Note: User must test G.ERROR on return if emptying buffer should cause a problem

TKSIZE

Purpose: To designate size of characters

Declaration: Procedure TKSIZE;

Procedures Called: None

Description: The common variable G.SIZE is tested and the user's buffer is loaded with the appropriate commands. G.SIZE has a value of large, medium, small or xsmall

Maximum Buffer Words: 2

TKLINETYPE

Purpose: To designate type of line

Declaration: Procedure TKLINETYPE;

Parameters: None

Procedures Called: None

Description: The common variable G.LINETYPE is tested and the user's buffer is loaded with the appropriate commands

Maximum Buffer Words: 2

TKSHIFT

Purpose: To shift alpha cursor in given direction

Declaration: Procedure TKSHIFT;

Parameters: None

Procedures Called: None

Description: The common variable G.DIRECTION is tested and the buffer is loaded with the appropriate command. G.DIRECTION has a value of up, down, left or right. Mode is set to alpha

Maximum Buffer Words: 2

TKGRID

Purpose: To draw a grid in given display area

Declaration: Procedure TKGRID(VAR K:INTEGER;NDIVX,NDIVY:REAL; VAR INTERSECT: ARRAY[1..3,1..?] of INTERGER);

Parameters:

Input

NDIVX	Number of divisions in X direction
NDIVY	Number of divisions in Y direction

Output

K	Number of intersection points
INTERSECT	Two dimensional array containing the intersection number, the X coordinate and the Y coordinate

Procedures Called: TKMOVE, TKDRAW

Description: Grid is drawn in display area defined in TKSTUP. The intersection points of the grid are saved in array intersect where:

Intersect[1,I] contains the intersection number
Intersect[2,I] contains the X coordinate
Intersect[3,I] contains the Y coordinate

Maximum Buffer Words: 9*NDIVX + 9* NDIVY

TKGRD

Purpose: To draw a grid without saving intersection points

Declaration: Procedure TKGRD(NDIVX,NDIVY: REAL);

Procedures Called: TKMOVE,TKDRAW

Parameters: NDIVX Number of divisions in X
NDIVY Number of divisions in Y

Description: A grid is drawn to user's specifications. No intersection points are saved.

Maximum Buffer Words: 9*NDIVX + 9*NDIVY

TKSTRING

Purpose: To write a string of ASCII characters

Declaration: Procedure TKSTRING(STRING:PACKED ARRAY[1..?] of CHAR; LENGTH : INTEGER);

Parameters:
Input
STRING Characters to be displayed
LENGTH Length of string

Procedures Called: SVC$

Description: Mode is set to alpha and the characters in string are sent to the terminal. This procedure uses it's own SVC call and no provisions are made to store the information on the graphic output file

Maximum Buffer Words: 1 + length of string

TKNUMBER

Purpose: To write a number on line below given X and Y

Declaration: Procedure TKNUMBER(NUMB,X,Y:INTEGER);

Parameters:

Input		
	NUMB	Number to be written
	X,Y	Location of point to be labelled
Output		None

Procedures Called: TKMOVE,TKSHIFT,TKSEND

Description: Cursor is moved to given X,Y. Alpha cursor is shifted down one line. Mode is set to alpha. G.SIZE is tested to determine size of number. This is used to label a point drawn on the screen. The shift down prevents the label from being placed on the point.

Maximum Buffer Words: 8 + number of digits in number

CONCLUDING REMARKS

TEKLIB is a general purpose two-dimensional graphic library of procedures written to interface between the TI 990 and an interactive terminal. The use of this library enables a user to input data graphically to the Finite Element Machine by way of the controller and to display output.

SAMPLE PROBLEM

```
PROGRAM SELECT;
(* THIS PROGRAM WILL DRAW THE FOLLOWING ACCORDING TO USER'S INPUT *)
(*      1. LINE
        2. RECTANGLE
        3. CIRCLE
        4. BULLET
        5. STRING
        6. NUMBER                                                    *)
 VAR    C,X,Y,X1,Y1 : INTEGER;
        X2,Y2,X3,Y3 : INTEGER;
        XDIST,YDIST : INTEGER;
        NUMBER,DELX,DELY : INTEGER;
        XL,YL : REAL;
        NDIVX,NDIVY,DIVX,DIVY: REAL;
        ANSWER : CHAR;
        STRING: PACKED ARRAY[1..12] OF CHAR;
        STRING1:ARRAY[1..12] OF CHAR;
        DELXR,DELYR,RADIUS,ANGLE,DELTA : REAL;
        NS,IPIE: INTEGER;
COMMON G: RECORD
         ENTRY : BOOLEAN;  (* SET TO TRUE TO REWRITE GROUT FILE *)
         XMINL,XMAXL,YMINL,YMAXL: REAL;(*   LOGICAL COORDINATES      *)
         XMIN,XMAX,YMIN,YMAX : REAL;   (*   PHYSICAL COORDINATES     *)
         XORG,YORG,XSCALE,YSCALE : REAL;
         ICX,ICY : INTEGER;            (* CURRENT RASTER POSITION    *)
         PTR  : INTEGER;               (* CURRENT BUFFER POSITION    *)
         TRAN : (SAVEON,SENDON,BOTH);  (*  SAVE, SEND,  BOTH         *)
         MODE : (GRAPHIC,ALPHA);       (* 29 GRAPHIC, 31 ALPHA       *)
         BEAM : (ON,OFF);              (* ON FOR DRAW, OFF FOR MOVE  *)
         INTENSITY: INTEGER;
         DIRECTION : (UP,DOWN,LEFT,RIGHT);
         SIZE : (LARGE,MEDIUM,SMALL,XSMALL);
         LINETYPE : (SOLID,DOTTED,DOTDASH,SHORTDASH,LONGDASH);
         ERROR : INTEGER;
         LBUFF: INTEGER;               (* LENGTH BUFFER DIMENSIONED  *)
         BUFF : ARRAY[1..80] OF INTEGER;(* USER BUFFER               *)
         TBUFF: ARRAY[1..40] OF INTEGER;(*TRANSMISSION BUFFER        *)
        END;
ACCESS G;
PROCEDURE TKRECT(DELX,DELY:REAL);EXTERNAL;
PROCEDURE TKARC(RADIUS,ANGLE,DELTA:REAL; NS,IPIE:INTEGER);EXTERNAL;
PROCEDURE TKCHEK(NWDS: INTEGER); EXTERNAL;
PROCEDURE TKCURSOR(VAR X,Y,CHARAC:INTEGER); EXTERNAL;
PROCEDURE TKDM(X,Y:INTEGER); EXTERNAL;
PROCEDURE TKDRAW( X,Y:INTEGER); EXTERNAL;
PROCEDURE TKERASE; EXTERNAL;
PROCEDURE TKLTP(VAR XP,YP: INTEGER; XL,YL: REAL); EXTERNAL;
(* TKNUMBER SHIFTS DOWN A LINE - USED TO LABEL A POINT *)
```

```
PROCEDURE TKNUMBER(NUMB,X,Y:INTEGER); EXTERNAL;
PROCEDURE TKNO(NUMB,X,Y:INTEGER);EXTERNAL;
PROCEDURE TKPTL(VAR XL,YL: REAL; XP,YP: INTEGER);EXTERNAL;
PROCEDURE TKSCA; EXTERNAL;
PROCEDURE TKSEND(NWDS:INTEGER);EXTERNAL;
PROCEDURE TKSTUP;EXTERNAL;
PROCEDURE TKSHIFT; EXTERNAL;
PROCEDURE TKSIZE; EXTERNAL;
PROCEDURE TKVECTOR(X1,Y1,X2,Y2:INTEGER);EXTERNAL;
PROCEDURE TKPOINT(X,Y:INTEGER); EXTERNAL;
PROCEDURE TKGRD(NDIVX,NDIVY:REAL); EXTERNAL;
PROCEDURE TKMOVE(X,Y:INTEGER);EXTERNAL;
PROCEDURE TKSTRING(STRING: PACKED ARRAY[1..?] OF CHAR;
          LENGTH : INTEGER);EXTERNAL;
BEGIN (*  MAIN   *)

(* INITIALIZE *)

  TKSTUP;
  G.SIZE := SMALL;
  TKSIZE;
  RESET(INPUT);
  REWRITE(OUTPUT);
(* DEFINE MINIMUM AND MAXIMUM X AND Y LOGICAL VALUES *)
  G.XMINL := 0.0;
  G.XMAXL := 100.0;
  G.YMINL := 0.0;
  G.YMAXL := 100.0;
  TKSCA;
(* DEFINE NUMBER OF DIVISIONS IN GRID    *)
  NDIVX := 1.0;
  NDIVY := 1.0;
 TKERASE;
(* COMPUTE VALUE OF EACH DIVISION *)
 DIVX := (G.XMAXL-G.XMINL) / NDIVX;
 DIVY := (G.YMAXL-G.YMINL) / NDIVY;
 TKGRD(NDIVX,NDIVY);
 WRITELN('   1 LINE');
 WRITELN('   2 RECTANGLE');
 WRITELN('   3 CIRCLE');
 WRITELN('   4 BULLET');
 WRITELN('   5 STRING');
 WRITELN('   6 NUMBER');
 WRITELN('   9 TERMINATES');
(* X3 AND Y3 CONTROL POSITION OF CURSOR FOR WRITELNS *)
 X3 := 10;
 Y3 := 650;
 C := 1;
 WHILE C <> 9 DO
 BEGIN
 TKMOVE(X3,Y3);
 WRITELN('SELECT');
 READ(C);
```

```
 CASE C OF
1: BEGIN (* LINE *)
   WRITELN(' INPUT BEGINNING');
   TKCURSOR(X,Y,C);
   Y3 := Y3-40;
   TKMOVE(X3,Y3);
   WRITELN(' INPUT END ');
   TKMOVE(X,Y);
   TKCURSOR(X1,Y1,C);
   TKVECTOR(X,Y,X1,Y1);
   END;

2: BEGIN(* RECTANGLE *)
   WRITELN( 'INPUT LOWER LEFTHAND CORNER ');
   TKCURSOR(X,Y,C);
   Y3 := Y3-40;
   TKMOVE(X3,Y3);
   WRITELN('INPUT UPPER RIGHTHAND CORNER');
   TKCURSOR(X1,Y1,C);
   DELX := X1-X;
   DELY := Y1-Y;
   (* CONVERT TO LOGICAL COORDINATES*)
   TKPTL(DELXR,DELYR,DELX,DELY);
   TKMOVE(X,Y);
   TKRECT(DELXR,DELYR);
   END;

3: BEGIN(* ARC *)
   WRITELN(' INPUT CENTER');
   TKCURSOR(X,Y,C);
   Y3 := Y3-40;
   TKMOVE(X3,Y3);
   WRITELN(' INPUT RADIUS');
   TKCURSOR(X1,Y1,C);
   XDIST := ABS(X1-X);
   YDIST := ABS(Y1-Y);
   IF YDIST > XDIST THEN RADIUS := YDIST
                    ELSE RADIUS := XDIST;
   ANGLE := 0.0;
   DELTA := 360.0;
   IPIE := 0;
   NS := 20;
   TKMOVE(X,Y);
   TKARC(RADIUS,ANGLE,DELTA,NS,IPIE);
   END;
4: BEGIN(* BULLET *)
   WRITELN(' INPUT LOCATION ');
   TKCURSOR(X,Y,C);
   TKMOVE(X,Y);
   TKARC(6.0,0.0,360.0,10,0);
   END;
```

```
 5:  BEGIN (* STRING *)
     Y3 := Y3-40;
     TKMOVE(X3,Y3);
     WRITELN( 'INPUT LOCATION');
     TKCURSOR(X,Y,C);
     Y3 := Y3-40;
     TKMOVE(X3,Y3);
     WRITELN('INPUT 12 CHARACTER STRING ');
     IF EOLN THEN READLN;
     FOR I := 1 TO 12 DO
     READ(STRING1[I]);
     PACK(STRING1,1,STRING);
     TKMOVE(X,Y);
     TKSTRING(STRING,13);
     END;
 6:  BEGIN(* NUMBER *)
     WRITELN(' INPUT LOCATION');
     TKCURSOR(X,Y,C);
     Y3 := Y3-40;
     TKMOVE(X3,Y3);
     WRITELN(' INPUT NUMBER');
     IF EOLN THEN READLN;
     READ(NUMBER);
     TKNO(NUMBER,X,Y);
     G.SIZE := SMALL;
     END;
 9:  BEGIN
     WRITELN('TERMINATED');
     END
OTHERWISE
     WRITELN('ILLEGAL CHOICE');
END; (* END CASE *)
Y3 := Y3-55;

END;
(* RESET SIZE OF CHARACTERS *)
G.SIZE := MEDIUM;
TKSIZE;
END.
```

APPENDIX

Buffer and File Management

The user's buffer of 80 characters in length is used to store the transmission code to be sent to the graphics terminal. This user's buffer is located in the common block and is labelled G.BUFF.

The common variable G.TRAN can have the value SAVEON, SENDON, or both. When G.TRAN is set to SAVEON, the graphic command is stored in the user's buffer (G.BUFF) and not sent directly. When the buffer is full the commands will be sent. When G.TRAN is set to SENDON the commands are sent directly. This option should be used when the user wishes to see the immediate effect of a command and also should be used for the last command in a given sequence of calls or of a program. The procedure TKCHEK can be used to test for overflow by the user. However, all the routines call on TKCHEK and will automatically empty buffer when the limit has been reached. The user should be cognizant of this feature when building a save file. Some routines depend on rontines that have been called before as described in the writeup and care should be taken that the buffer is not emptied between these routines.

When the value of G.TRAN is set to both, the graphics data will be sent to the screen and also written on the text file 'GROUT'. The user must assign the synonym 'GROUT' to a save file before a session or copy it to a save file after the session if he wishes to bring back the display at a later date.

```
   1 LINE
   2 RECTANGLE
   3 CIRCLE
   4 BULLET
   5 STRING
   6 NUMBER
   9 TERMINATES
SELECT

INPUT LOCATION

INPUT 12 CHARACTER STRING


SELECT
 INPUT BEGINNING
 INPUT END


SELECT
INPUT LOWER LEFTHAND CORNER
INPUT UPPER RIGHTHAND CORNER


SELECT
 INPUT CENTER
 INPUT RADIUS


SELECT
 INPUT LOCATION
SELECT
 INPUT LOCATION
 INPUT NUMBER


SELECT
```

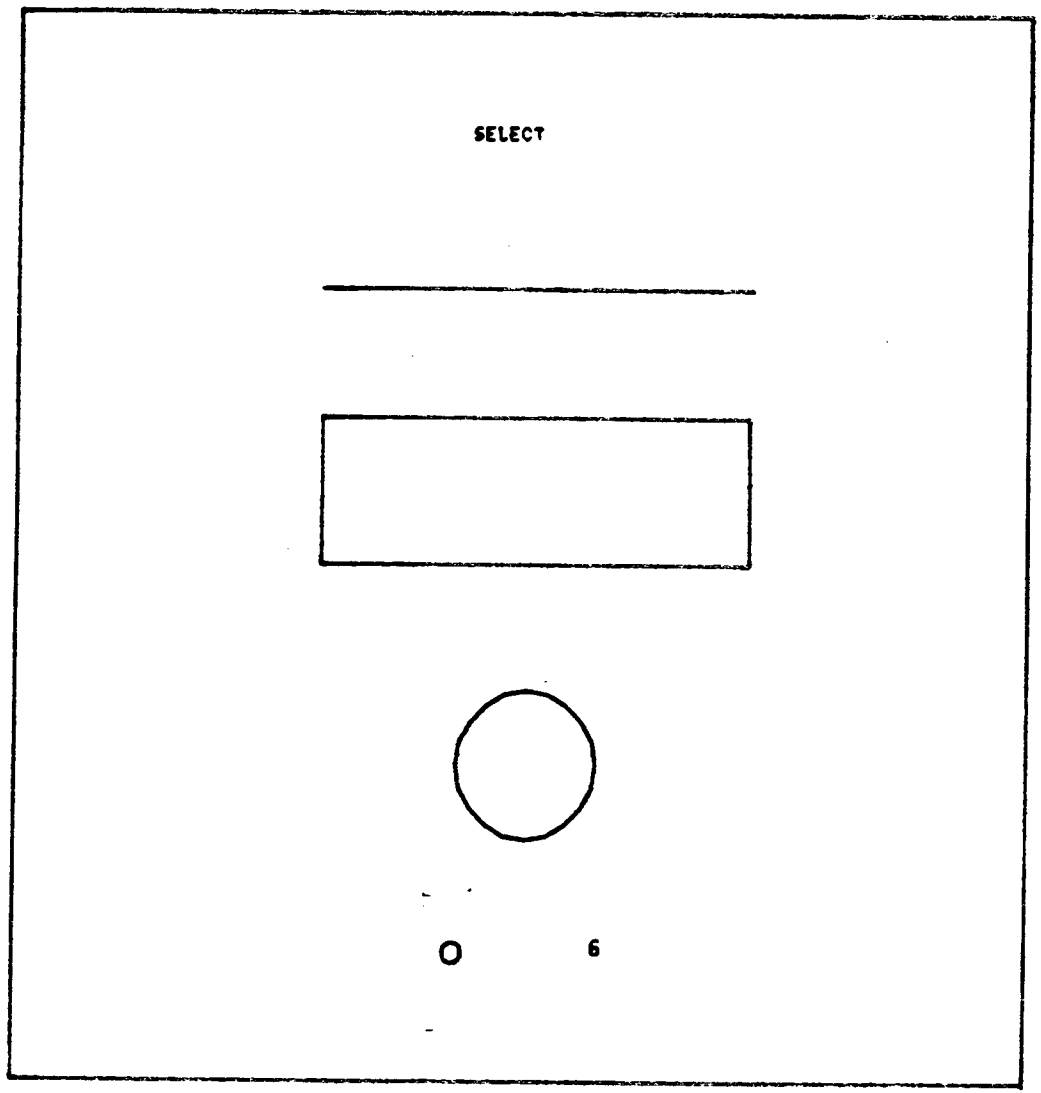

Figure 1. - Example of output from program SELECT.

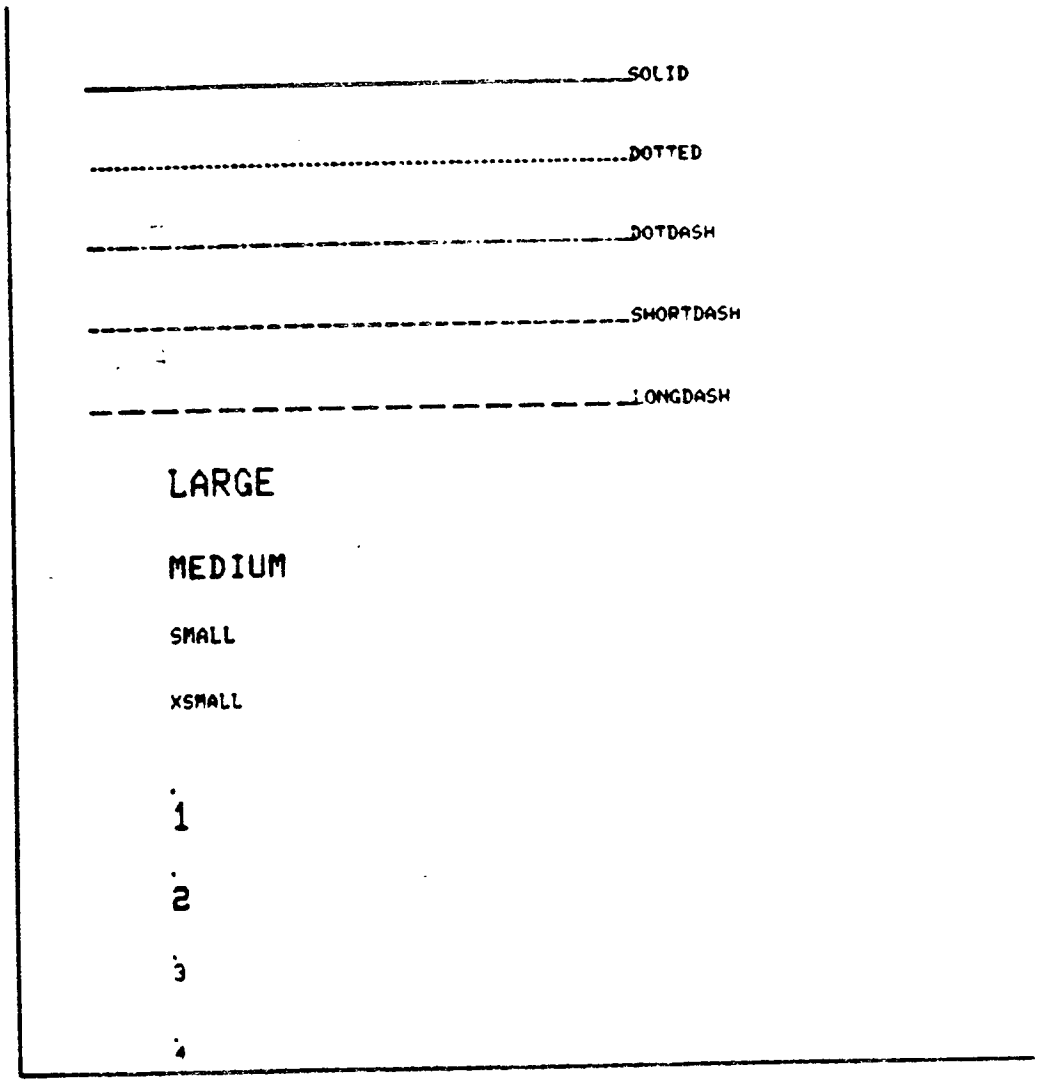

Figure 2. - Example of LINETYPES and SIZES.

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA TM-84633		

4. Title and Subtitle	5. Report Date
TEKLIB Graphics Library	March 1983
	6. Performing Organization Code: 505-37-13-01
7. Author(s): Susan W. Bostic	8. Performing Organization Report No.
	10. Work Unit No.
9. Performing Organization Name and Address: NASA Langley Research Center, Hampton, VA 23665	11. Contract or Grant No.
	13. Type of Report and Period Covered: Technical Memorandum
12. Sponsoring Agency Name and Address: National Aeronautics and Space Administration, Washington, D.C. 20546	14. Sponsoring Agency Code

15. Supplementary Notes

16. Abstract

TEKLIB is a library of procedures written in TI PASCAL to perform basic graphic tasks. TEKLIB was written to provide an interface between a graphics terminal and the TI 990. The TI 990 is used as a controller for the Finite Element Machine which is an array of microprocessors designed to solve problems by finite element methods in parallel. The use of TEKLIB provides a means of inputting data graphically and displaying output.

17. Key Words (Suggested by Author(s))	18. Distribution Statement
Graphics Computer software Computer routines	Unclassified - Unlimited Subject Category 61

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	25	A02

For sale by the National Technical Information Service, Springfield, Virginia 22161